What is claimed is:

1. A procedural computation engine for generating and serving executable high-level code comprising:

a Graphical User Interface for creating procedural computation schemas;

a parser for interpreting output from the Graphical User Interface;

a compilation component for hierarchal node-structuring of data; and

a server component for providing access to generated information;

characterized in that a programmer operating through the Graphical User Interface pre-creates at least one procedural computation schema including the algorithmic function or functions and input needed to produce computational results, the data of the schema output as a markup file interpreted by the parser and in cooperation with the compilation component generates an executable computation model accessible and executable through the server component.

2. The computation engine of claim 1 wherein the Graphical User Interface is of the form of an interactive spreadsheet processing application and the computation model is a rating model.

3. The computation engine of claim 1 wherein the parser is adapted to read XML and to write in Java Document Object Model structure.

4. The computation engine of claim 1 wherein the compilation component includes a lexical scanner and a code generator.

5. The computation engine of claim 1wherein the computation models are rate models pre-stored for access by the server component upon request over a network connection.

6. The computation engine of claim 1wherein the model is a rate model designated as a user function to be embedded in another rate model.

7. The computation engine of claim 1wherein the computation models are rate models and a knowledgebase configurator has access to the stored rate models through one of remote method invocation or through remote call procedure over a network connection.

8. The computation engine of claim 5 wherein the network connection is one of an Internet or an Intranet connection.

9. The computation engine of claim 7 wherein the network connection is one of an Internet or an Intranet connection.

10. The computation engine of claim 2 wherein the processing application can interpret Extensible Markup Language and can save data in the form of Extensible Markup Language.

11. A rating service comprising:

a procedural computation engine having a graphical user interface for creating procedural rating schemas; a parser for interpreting output from the graphical user interface; a compilation component for hierarchal node-structuring of data; and a server component for providing access to generated information;

a knowledgebase configurator for configuring service requests; and

a software interface application through which requests for rating are submitted;

characterized in that an end user accesses the configurator through the interface application and submits parameters for configuration of a service request whereupon the configurator calls the server component of the computation engine and selects a rate model from a pool of rate models that fits the request parameters, the rate model applied to and executed within the configuration model to produce the rating results through the application interface.

12. The rating service of claim 11 wherein the software interface application is an insurance application suite.

13. The rating service of claim 11 wherein the parser is adapted to read XML and to write in Java Document Object Model structure.

14. The rating service of claim 11 wherein the configurator is a Web-based configurator and calls the server component of the computation engine using one of remote method invocation or remote call procedure.

15. The rating service of claim 11 wherein a service configuration contains more than one rate model, the models individually executed according to optional scenarios.

16. The rating service of claim 11 wherein a service configuration contains more than one rate model, one rate model designated as a user function embedded in another rate model.

17. The rating service of claim 11 integrated with a software framework for enabling client security verification, user interface generation, workflow management, database search functionality, and language transformation for presentation to alternate platforms and interfaces.

18. A method for modeling a procedural rating schema comprising steps of:

(a) using a spreadsheet interface, inputting at last one algorithm or formula into a specific cell or cells for rate calculation;

(b) in the same interface, entering the required input values into cells and marking the cells that will carry the input values into the calculation;

(c) in the same interface, marking the cell or cells that will show the output results from the calculation; and

(d) saving the data as a rating computation file in Extensible Markup Language.

19. The method of claim 18 wherein in step (a) the spreadsheet calculation functionality includes one or more calculation functions Loop Add, Loop Multiply, Loop And, Loop Or, and Loop Append.

20. The method of claim 18 including a step between step (a) and step (b) for inputting optional parameters for optional executions according to the differing parameters.

21. A method for compiling a rate model from a rate computation file created and saved in Extensible Markup Language comprising;

(a) parsing the file in a parser;

(b) building a symbol table from parsed elements;

(c) outputting a Java Document Object Model file from the parser;

(d) building a syntax tree from the Java Document Object Model file;

(e) validating the hierarchy of the syntax tree;

(f) generating model code for executing the file; and

(g) outputting the executable component.

22. The method of claim 22 wherein in step (a) the parser reads Extensible Markup Language and writes Java Document Object Model language.

23. The method of claim 22 wherein step (b) is referenced when step (d) is executing to access symbols from the symbol table.

24. The method of claim 22 further including a step between step (e) and step (f) for optimizing the built hierarchy.

25. The method of claim 22 further comprising a step between step (f) and step (g) for compiling the data into alternate object model languages for presentation across platforms.

26 The method of claim 18 wherein in step (a) the spreadsheet calculation functionality includes XML operation functions get value and set value.

27. The method of claim 21 wherein in step (d) the source node of the tree is a formula node with a condition variable of true or false.

28. The method of claim 26 wherein variables are scoped using block translation according to the condition type of true or false.

29. The computation engine of claim 4 wherein the compilation component includes at least one block translator for scoping variables.

30. The computation engine of claim 4 wherein the compilation component creates loop constructs to resolve variables in the case of a dynamic query, the loop calculations performed to create a formula.